Cover Slide

(1)

"in quotes" – like all software projects, ~~its~~ it is behind schedule

Within a couple of months I ~~hope~~ expect to be able to

~~remove the quotes~~

Summary:

What do I ~~mean~~ by a virtual Unix? A collection of routines

running on top of Tops-20.

vs 11/70, EUNICE

~~Context~~ 1) Describe what I mean by a virtual Unix } different kinds

2) ~~Discuss~~ Mention some of compiler issues

3) Runtime

~~What do I mean~~

What did this project consist of? – Notice NO ~~tense~~ specified!

(2)

1. Compiler

I ported the "Port C compiler" (the same one as is used on the Vax + many other machines) &

~~I'll discuss~~ This is ~~certainly more than~~ at least 1/2 the work, + maybe more.

I will ~~discuss~~ mention some problems ~~also~~ in a minute.

2. The Providing V7 Unix Environment –

~~A)~~ provide ~~many~~ most of the C library routines as just that, linkable object modules. ~~Many~~ Many section 2 system calls are also provided.

↓ ~~another~~

B Pseudo-Kernel ~~of this~~ providing Unix section 2 calls.

For ~~diff~~ ~~equal~~ reasons, ~~Because of~~ ~~...~~ I am putting some oft-used section 3 routines here also, like printf.

~~I will discuss this in ...~~ I'll say more about this later.

3. Porting user programs

If the first 2 are done right this 3rd step should be trivial – and in most cases it is. ~~There are also~~ ~~There are some~~ restrictions of the Tops-20 environment which create a few exceptions – I'll mention the exceptions later.

2 hrs } 5:15 – 9:00

1 1/2 lunch

3 1/2

-30 permits

-30 drive + park

-~~...~~ travel

eg UNIVAC - UNICRAP

WHY NOT just Unix?

(3A) WHY ~~Un~~ PSEUDO-UNIX on TOPS-20,

- ~~Pretty much~~ Both TOPS-20 + Unix are "NICE". You'll pretty ~~much have~~ take on faith, but - for example
  - COMND JSYS
  - General page-mapping - from Tenex
  - File Version #'s.
  - 1/34 ~~Frontend - Backend~~ keeps each other alive - soft failures

WHY NOT just TOPS-20?

- All the good things we know + love - Motherhood
- C-SHELL ; all the good features - redirection, pipes, Berk job control
- utils - grep, make, yacc, gosling's EMACS

  IN C! not Macro

(3B) ~~GOALS~~

GOALS

- ~~Access to Both Unix + T20~~ (good things)
- ~~Usable from B~~ SAME FILE FMT for Unix + T20. (editor)
- ~~Files must be accessible both from pseudo Unix~~
- Access to both Unix + Tops20 environment.
- Usable ~~under T20~~ w/ and w/o shell (eg directly under EXEC) with ~~a lot~~ some of the good things from the shell, ie redirection + globbing
- Keep the load modules small (eg Unix - lots of
- User program must be insensitive to "Kernel" changes.
- No Hacking of TOPS-20 ~~kernel~~ or "shell" (mario wouldn't want to)

5Jd (3) <u>Motivation</u>

(1) ~~[illegible]~~ A systems programming language for 20 -

(2) <u>Common</u> sys prog language - 1 20, 5 Unix systems

No fun writing twice in BCPL + once in C. for this need ~~[illegible]~~ Unix environ.

(3) ~~Port Unix softw~~ language with extended addressing support

(4) Port Unix software - This grew on us with ~~[illegible]~~ -

→

---

(4) So - that's ~~[illegible]~~ why we did it — Now How we did it. —

Quite some time ago - I'll keep you mercifully ignorant of the exact date — ~~will let the exact date remain merciful~~

I was ~~"chosen"~~ given the great honor of this project - Now why me? I asked, and other people certainly asked that, too, after a short while

For example - I didn't not know ~~the~~ any of these:

(1) PDP-11 architecture

(2) Unix -

(3) PDP-10/20 architecture

(4) Tops-20

(5) C

Why'd they do this? The only way they could get someone to learn this + then sucker him into spending their next year (after he'd finally learned it) recruiting our Unix systems! — ~~[illegible]~~

So, armed w/ ~~and~~ this vast storehouse of ignorance, I blithely stepped in —

~~[illegible]~~ <u>BOOTSTRAPPING HOSTS</u>

(5) (1) 11/34 V6 Unix

(~~both~~ pcc + yacc which its use on split I/D V7 progs) - all we had.

Actually did get some useful work done.

(2) 11/45 V6 Unix

(3) 11/45 V7 Unix + I could start to get some work done —

But there was a lot of wasted energy in there

LINES OF ASM

(6) C Compiler issues

- ~~[illegible]~~ QUICKIE
PDP-10

WORD-ADDRESSABLE

- 36 bit word, any size byte accessed ~~[illegible]~~ indirectly
via a "byteptr"

std size 7 - 5 per word.

I Mark Horton: ~~#~~ 36 w/o 7 ~~#~~ 1.

- C "spec" problems — sizeof = size in bytes -
- compiler doesn't like it Assumes LONG > INT
- WORSE - ~~C library~~ STDIO library + std practice
assumes will get everything if copy all the bytes —

⇒ - 9 bit bytes -

turns out not an ~~[illegible]~~ op sys interface problem.

(~~4~~) ~~After struggling~~

After struggling w/ the compiler for some time,
SAM LEFFLER ported PCC to a word-add.
machine + also wrote a detailed doc

(7) FIRST PGM 2 copies

HACKED Unix for 8 months
Pushed others on hold.

(8) LIBRARY

- ~~Why~~ ~~Usually~~ ~~often~~ Usually people re-write stdio for another machine when write a C compiler for it. Lots of ~~[illegible]~~ utils don't use stdio, tho. aside from shell, etc.

- OPEN, CLOSE, READ, WRITE, MALLOC, (SBRK), FREE, ISATTY - get almost ALL stdio
~~Why~~ Struggle to Emulate the semantics "exactly" - all rest follows

- Why write it when you can "borrow" it? euphemism ~~[illegible]~~ lawyers here.

(9) V UNIX -

- RTS ~~[illegible]~~ lives in locore, includes globbing + redirection parsing.
- Pgm looks for flag, RTS not there it loads it. ~~&~~ + RTS does smile line interpretation
Otherwise, no special processing

- ~~She~~ ~~Fork~~ New process - RTS shared memory, includes data ~~of [illegible]~~ from proc + used structures - Pipes are shared pages.

- ~~Not~~ Borrow kernel code almost straight - ufalloc, falloc to alloc fd's + files - pipe.c - almost ~~just~~ just emulate wakeup + sleep except that uses interrupt MASKING

(10)

Unix on TOPS20

(1) ~~what~~ Similarities

1. files are byte-streams
2. ~~Dir~~ Filesystem is a directory tree
3. current directory, but different semantics

(2) Diffs

1. Process structure - strict hierarchy
   PIDs are only relative
2. CRLF vs LF — fseek re-write - BITCH
   ~~Ex~~ MUNDANE.
3. ~~directories unreadable~~
4. Directory size quotas, subdir quotas
5. NO LINKS } explain why need
6. Long filenames }
7. ~~cannot~~ "connect" semantics

(3)

=> Would be easier to port Unix itself — filesys + process struct.

Most of these problems go away

~~But~~ But then no access to 20 pgms

(11) Current Routines

To wow you.

Note all stdio (~~fseek~~ fseek + magic cookie restart applies)

bsrni [?] ioctl - bitch

FORK/EXEC

Floating Pnt

jsys - lines of ASM - ~120

- JSYS
- ESV, CRET
- SETJMP/LONGJMP
- FORK/EXEC helper
- ~~INIT~~ STACK INIT

(12) Pgms running NOW

- temacs
- Rogue!

} ~~Emulated~~ NO substantive MODS. ~~NO MODS~~

(13) Portability problems in retrospect

(14) Lessons learned

AVAIL

(15) 6 pack sweepstakes

AVAILABILITY

contact me
or we have
our license agreement here
like Berkeley

Q's (- Shangri La)
FRANK

6 parks in Utah

Morals ~~Lessons~~ in Retrospect

1- don't waste time ~~on struggling with a system incapable~~
~~of trying to make a s~~
struggling with a hopelessly inadequate support environment (unless you already know a lot — now I would go get one of the many user-friendly facilities)

2- ~~Need~~
~~These are~~
~~Do your damnedest to find~~
Get real lucky & find out what other people have done along same lines
e.g. Steffler

3- ~~This~~ Classic: ~~Need Time to get an~~
Bootstrapping: Time to complete $\approx \frac{1}{e^{FTP\ time}}$

Shared disk - Still ~~very~~ useful when porting programs.

4. Don't waste time at beginning worrying too much about picky ~~s~~ details of C semantics - e.g. the C spec is ~~also~~ not ~~a~~ much of a "spec", for it is frequently ambiguous ~~all~~ - Many places it implicitly assumes long > int.
e.g. UNSIGNED2 - don't ~~matter~~ as far as porting programs, cause 36 > 32!
And working out Code Sequences

5. Do spend time working out the calling convention.
Be conservative - Can change later for speed, perhaps.

Issues ~~that~~
1- Must be able to handle var # of args.
2- Keep info in stack ~~like~~ useful for debugging - don't squeeze every last word out.

Porting pcc

6. When you've got 16 regs, don't worry ~~about~~ too much about the fine points of estimating register use, which is ~~twice~~ big emphasis in pcc.

→ Do use pcc cause it's defines the language.

Didn't know anything about –

11/34 V6
11/45 V6
11/45 V7

① – PDP-11

well, ~~you~~ say, that's no problem, after all you're just writing a PDP 10/20 compiler. But – ~~[illegible]~~ base compiler was for PDP-11 (with this caveat: "   "), and ~~[illegible]~~ of the 11 was implicit for real understanding of the C spec.

well that ~~[illegible]~~ one's not so bad –

③ – PDP-10 k –

well this ~~gets~~ one's a little ~~harder~~ more of an obstacle, But it's just another machine, right?

④ – Tops-20

Just an op-sys, & after all PCC just generates ASM, & since I ~~was~~ am now nearly an expert on 10 machine language, that's no problem –

② – Unix –

Just cause ~~that was the environment~~ I was trying to ~~[illegible]~~ get (bootstrap) a ~~[illegible]~~ split I/D V7 kernel on an 11/34 using V6, and trying to emulate them 100% faithfully .. But after all, I ~~[illegible]~~ wouldn't be using on a PDP-11 Unix system, I'd be using on a PDP-10 – But

⑤ – C –

! Just another language, right – what better way to really learn its semantics than study the compiler – Hell, I can define the semantics!

Why'd they have me do this? The only way they could → for writing the compiler get ~~someone~~ who knew all the above + ~~[illegible]~~ then send him into spending the next 18 months (after he'd finally learned this) writing on various Unix systems!

~~Really — take Hackers~~ ??

Goals — No hacking of Tops-20 — Exec or Monitor

Need talk about

- Why me? Relate ignorance
- Describe KL10,
- Why Unix on Tops-20 — say why 20 is nice
- What diff/same about Unix/Tops-20 — what ease/trouble Bksh.
- (1) Background Motivation? D1800, Tops-20 wimp; 3) Extended Address; 2) SYS — programming lang; 1) port software — 5 Unix Sys & 1 20
- Rationale for V unix vs library — Pgm Size/ Why several of —
  open, r, w, malloc — got about all of stdio
- ~~Lift~~ "Bottom code" ⇒ not only lift ~~...~~ stdio, but kernel — malloc, falloc.
  — interrupt masking issue
- Describe how maintain access to Tops-20 stuff, eg pgm bits to see if loaded.
  sharing by RTS
- Outline process structure — shared memory
- Easier to port Unix itself — eg fibs, proc structure

Review development history

- ~~Pcc~~ — C ~~....~~ spec issue — How evenly divisible words
- PCC — Leffler, comes together (2 passes in 1)
- LINES OF ASM
- 1st pgm

96 — ~~ROUTINES I have now~~ — Note jsys

- ~~Describe LINK?~~
- ~~TO~~

98 — Portability problems — Externs, long extern names, upper/lower case mapping
CHAR c = getchar(), Seek, Varargs
why C is good for port: — importance of typing for portability, eg BCPL
— CPP: long constants, x-compiling

97 — Pgms running NOW — + termcap
Rogue!

99 — Lessons learned

—

— 6 people to choose of name — yes Utah dubs it.
I have a good one but 2 century. B I and — come Richer

CRAP

$512 \quad 5\times10^2 \times 2\times10^3 = (10^6)$

$\frac{512}{5} \quad 2560$

512 pages / 2 / 1000,000 / 512 / 2K / 512 pages

42-381 50 SHEETS 5 SQUARE
42-382 100 SHEETS 5 SQUARE
42-389 200 SHEETS 5 SQUARE
Made in U.S.A.
NATIONAL

Lives of ASM/C

talk on portability libr, patching filesys.

fresh Tops-10

Usenix talk — 6 people to whom organizat of the Ghosts Now — one sole pledge, yes, we finish it in Utah — cutting like UCB

— Motivation — B1800! Tops-20 for wanup — now, 2 years later....

—

— 1) SYSTEM pgms/UTILS ~~in C + ?~~ Both in C + BCPL

2) Nice sys-progmg language on 20

3 parts

(A) 1) ~~Compiler~~

(B) { 2) ~~C~~ C-like library

3) ~~Virtual~~ Unix

4) porting User pgms

Do one by me — Unix, KL10, Tops-20 system, PDP-11 comparison (C)

cepiont of ~~pdp11~~

11/34 V6 Yacc/Cref

11/45 V6

11/45 V7

1st pgm!

1st ~~untie~~ — ~~putchar(x)~~ quota exceeded!

— Sam Leffler's tour.

Motivation for ~~RTS~~ Virtual Unix — 1) Pgm Size .EXE file — quotas on tops-20!

2) STDIO — why screw around with fopen, etc — when can just do section 2.?

- open, read, write, close
  get almost all stdio

- why screw w/ section 2 when can lift code straight from Kernel? (except locking issues)
  eg pipe.c, falloc, ufalloc.
  ~~just~~ if Unix ~~wasn't~~ ~~often~~ didn't use interrupt [locking], then would work as is. wakeup/sleep

- outline ~~for~~ process structure — shared memory vs ~~?~~ Unix process (inefficient)

— Maintain access to both Tops-20 + Unix — at use levels —
  ~~check for RTS ?~~ - ~~Sys~~ Exe Globbing done by RTS
  - Not load, load RTS, ~~?~~ + glob, otherwise no.

Easier to port Unix itself (e.g. process structure + filesys)

availability — because basically doing all the stuff in sys?.c. (??)

II

Pgms —
grep
Yacc lex
~~sed~~
games — Rogue
Termcap

Portability Problems
(Missing Externals) — get Berkeley & take the dungeon bonk!
~~Ext~~ long external names, upper/lower case mapping
CHAR c; — inefficient should be int. — int returned for putc, etc.
(Seek)
Varargs.h

TWENIX
~~OPEX~~ 10/21/81

How to ~~han~~ manage run-time support -

Goals

1. Keep .EXE files small

2. ~~Atom~~ Keep minimum run time image support reasonably small, e.g. be able to load part only.

3. Be able to run programs both with and w/o the shell. ie, environment must provide shell support fn's optionally

4. Keep loading time minimal.

5. Fast linkage.

(? 6. Write in op. sys. independent manner?)

7. Be able to move stdio routines to kernel ~~pkg~~ package 4 ease.

8.) Can't re-compile or re-link pgm when package changes.

Possibilities

1. Easy - loadable run time ~~support~~ package a la FOROTS

Include a dispatch vector with each program -

```
access   | JRST 3 | (RX)        PUSHJ access,
chmod    |      4 | (RX)
chdir    |        |
~~en~~   | ~~JRST 2~~ | ~~44~~    JRST
```

This works fine for one separate ~~loading~~ package. What about separate pieces? -

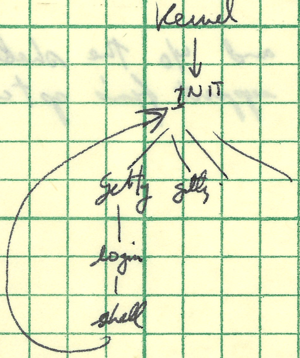
```
~~3(RX~~   RX --access-->  |   :
                           |
~~access~~ chmod -->       | ~~IF ...+1 to~~
                           | SKIPE .+2
                           | JRST <LOC>
                           | LA ~~...~~ CALL #
                           | ~~JRST LOADIT~~  PUSHJ, LOADIT
```

If dynamic loading, don't need initial dispatch vector

LOADIT - Get - looks at index in front of module & fills in those locs -

INDEX

syscall #	offset in module
ϕ	


Kernel
INIT
getty
getty
login
shell

Fork/~~exec~~ ~~exec~~ with RTS -

Fork probably no problem.
Exec - PUSH, execl
JMST X(RTSBASE)

~~Exec~~ ⋮

GET
Zero the stack -(parse agv, ~~argc~~ argc?)
B entry ~~vecto~~ start

How about shell vs no shell, job initialization, etc.

If no shell: Naked pgm 1) starts,
2) sees not initialized in JOBDA or RTS spot.
3) So loads RTS + branches to its init routine.
a) RTS ~~init~~ does 1 time starting stuff - sets up 0, 1, 2
[b) Parses argv, argc, re-direction]
c) ~~set up 0, 1, 2~~

↓
may work automatically cause shell removes > < from argv

If shell

(extended addressing)

Describe TOPS-20 -

Why not just Unix?

~~[illegible]~~ If you've seen TOPS-20 you'll just have to take on faith, but a few things:

- COMND JSYS
- Pages - Mapping - general scheme. / TENEX
- MM
- ~~logical names~~ SYSTAPY → Version #s on files
  Deleted stay around

Why not just TOPS-20?

All the good things we know + love; Motherhood

a) shell; ~~redirection~~ programmability, pipes, Berkeley job control, history, alias (sub?)

b) utils - grep, make, (gemacs?), ls

↓

(XSEARCH

in C - ~~a bit~~ not MACRO!

c) C

- khakis
- grey cords
- loafers
- sneakers
- jacket
- Brown sweater
- Wind breaker
- Slicker top
- ~~[illegible]~~ shorts
- bathing suit
- Frisbee
- Sailing gloves
- ~~[illegible]~~ T-Shirt

White Shirt
Pin Striped Shirt
Blue Striped Shirt

Wed · Thick Sweater
Thurs · White Socks
Fri

jeans ??

The preprocessor makes porting infinitely easier -
e.g. long constants 32 → 36 bits

Importance of typing (++p)

Extended Addressing

Critical calls -
open, ~~stat~~ read, write, malloc (sbrk)

CB, COMM, DC, EGREP, GREP, LEX, OD, SED, UNIQ, WC, YACC
(CPP, PCC)